Chapter 4
Organization and Agent Architecture in TRACE

Existing formalisms for implementing organizational policies assign specific roles to each agent in a multi-agent system. Examples are hierarchical organization, contract net protocol, social reasoning mechanism, and the use of matchmaker agents (described in Chapter 3). These policies allow the problem solving roles of the agents to change dynamically but do not adapt to variations in computational load on the multi-agent system. They are designed to operate for a predefined maximum problem-solving load and fail to respond, when the number of task requests, exceed this limit.

On the other hand a decrease in the problem-solving load will result in surplus resources. What we need is a mechanism for dynamic allocation of resources to the MAS to allow it to operate in unpredictable environments. Our objective is to find a suitable organization that exhibits high performance despite unanticipated changes in the environment (type and frequency of requests) and time constraints. In such a scenario there is no single organization that yields optimal performance under all conditions. What is therefore required is some way of having the multi-agent system dynamically change its organization so as to always match its resources with the problem solving demand. This requires changing the agents that comprise the organization and the organization structure. Our approach is to have agents as flexible entities, which can be dynamically restructured in response to changes in the environment. The knowledge that agents possess is also changed dynamically by migrating portions of it from agent to agent.

In this chapter and the next two, we propose an adaptive organizational framework called TRACE. It is assumed that agents obey the responsibility code of
conduct for joint activity as proposed by Nick Jennings [96]. The resulting framework meets the needs of soft real time applications where the computational load cannot be predicted, and also utilizes system resources efficiently.

Section 4.1 gives an overview of the proposed framework. Section 4.2 describes the MAS organization for TRACE. Agent architecture is described in Section 4.3. This is followed by the conclusions in Section 4.4.

### 4.1 Overview

The proposed framework consists of several problem-solving organizations where each organization is comprised of multiple agents that may be grouped into teams for specific problem solving. Problem solving requests with an associated priority and deadline arrive at the agents of these organizations. A request that arrives at an organization is solved cooperatively by agents within that organization and independently of the other organizations. The rate of arrival of problem solving requests at each of these organizations varies with time. As a result, the requirement for resources also varies. At any particular instant, some organizations may have surplus resources, while others have insufficient resources and thereby turn down problem solving requests. In order to minimize these lost requests, the allocation of resources to organizations needs to be changed dynamically. This reallocation results in reorganization of the multi-agent system and is intended to balance demand for resources at each organization with its supply ('resource' in this discussion refers to an agent).

The allocation of additional resources increases parallelism within the organization, resulting in an increase in the number of requests whose deadlines can be met. However, the total set of resources over all the organizations that constitute the multi-agent system always remains constant.
This generic framework can be used for realizing soft real-time applications where the problem-solving load on an organization varies non-deterministically. For example, applications, which require tasks like condition monitoring, fault detection, and diagnosis to be performed continuously, can be implemented using TRACE.

Following a layered approach, we divide the problem of developing an adaptive organizational policy into two broad sub-problems viz.

1. Allocation of tasks to agents within an organization and
2. Allocation of resources (agents) to each of these organizations

This division simplifies the design of agents

The multi-agent system organization and the individual agent architecture for TRACE are described in this chapter. The problems of task allocation within an organization and resource allocation to organizations are addressed in Chapters 5 and 6 respectively.

The multi-agent system organization that we propose facilitates reorganization to accommodate load variations and the agent architecture allows individual agents to exhibit team rationality and adapt to unpredictable changes in the environment

4.2 Multi Agent System Organization for TRACE

In TRACE, the multi-agent system is viewed as a collection of independent problem-solving organizations working under time constraints. The elements that constitute an organization are

- The agents
- The organization structure
- Types of tasks the organizations carry out

Each of these is described below
4.2.1 Agents

Agents can make decisions and take actions, and are constrained by their organizational role. For example, role may be that of a manager or a contractor. The actions of which the agents are capable depend on their capability and knowledge. An agent's knowledge is comprised of task-based knowledge and organizational knowledge, i.e., knowledge about other agents in the organization.

![Multi-agent system organization in TRACE](image)

**Figure 4.1** Multi-agent system organization in TRACE

Agents are of three types:

1. A fixed set of *permanent agents* (shown as blank circles in Figure 4.1) that an organization owns and that always belong to it. The number of agents in this set...
Chapter 4. Organization and Agent Architecture in TRACE

is the minimum number of agents that are required to complete the organization and always keep it in operation.

2. A set of **marketable agents** (shown as shaded circles in Figure 4.1) which each organization can access. The agents in this set are dynamically allocated and are allowed to enter or leave any of the organizations. The allocation of these agents is controlled by a special kind of agent called the **resource manager**. Every organization has an associated resource manager that keeps track of the resources required by it.

3. The **Resource Manager agents** (RM) set up markets for marketable agents and manage their buying and selling and thereby dynamically reorganize the multi-agent system. This is a role and the agent may additionally take up problem solving activity.

Initially it is the permanent agents of the organizations that process incoming requests. As problem-solving activity progresses, the resource managers periodically determine the current requirements of their organizations and use a market-oriented protocol to arrive at a suitable allocation of marketable resources to the organizations. This reallocation of resources results in a reorganization of the multi-agent system. Every organization therefore consists of a set of permanent agents together with zero or more marketable agents that carry out domain problem solving activity. The permanent agents play the role of managers and contractors but marketable agents only play the role of contractors. These roles will become clearer when we describe the task and resource allocation protocols in Chapter 5 and 6 respectively.

### 4.2.2 Organization Structure

The organization's authority and communication structure is described in terms of links among agents. In the authority structure, the links show who has authority over whom and thus reports to whom. In the communication structure, the links show who
Chapter 4 Organization and Agent Architecture in TRACE

talks to whom. Figure 4.2 illustrates possible authority and communication structures. In a collaborative structure all links are possible, in a hierarchy there is a central or apex agent. As the agents that we consider are autonomous, TRACE communication and authority structures are collaborative.

Authority Structure or Communicative Structure

Collaborative team                      Hierarchy

![Collaborative team structure](image1.png) ![Hierarchy structure](image2.png)

Figure 4.2 Organization Structures

4.2.3 Types of Goals

At any point of time the agents in an organization are engaged in executing goals. A goal may be composed of sub-goals with dependencies among them. Thompson [47] identified three such dependencies:

- **Pooled** The results from two or more goals are jointly needed to perform a different goal
- **Sequential** Two or more sub-goals must be performed in a specified sequence
- **Reciprocal** Two goals depend jointly on each other

TRACE handles all three types of dependencies. The set of goals faced by an organization can be thought of as its environment (or problem space). The type and frequency of requests (computational load) varies with time.
Agents strive to achieve two types of goals: those which can be undertaken by individuals (primitive goals), and those in which groups (at least 2) agents work together (social goals). So for example, if two agents collaboratively lift a table, then the goal 'lift table' is social because it involves a team of agents. Social goals ultimately give rise to primitive goals because only individuals have the ability to act. Thus social goal 'lift table' may give rise to primitive goals of agent 1 lifting at end 1 and agent 2 lifting at end 2. In the following discussion we use the term *task* or *action* to mean the sub-goals that are required to achieve a goal.

As the multi-agent system is assumed to operate in complex and dynamic environments, it is essential to ensure that agents exhibit team rationality and remain coordinated even when something unexpected happens. The agent architecture we propose therefore makes use of intentions to model collaborative multi-agent behavior.

### 4.3 Agent Architecture in TRACE

Figure 4.3 shows a high-level agent architecture for TRACH. The rectangles correspond to processes and ovals to data stores. This is a high level BDI architecture for collaborative multi-agent behavior in which intentions play a central role. Intentions are used both to coordinate actions (future directed intentions) and to control the execution of current ones (present directed ones). An agent has a local knowledge base that includes the following:

- **Beliefs**: Represent information the agent has about its current environment and are accessible to all the processes.

- **Desires**: Represent possible courses of action available to an agent.

- **Intentions**: Represent the agent's current focus; those goals that it has committed to bring about.
Chapter 4 Organization and Agent Architecture in TRACE

- **Joint Intentions** Represent the fact that a group of agents are jointly committed to a goal and the means for achieving the goal.

- **Recipe Library.** Contains goals and their associated recipes (plans). Recipes specify a set of steps (actions/tasks), with some temporal orderings that are necessary to accomplish a goal. Tasks are also associated with a lower bound on time that indicates the minimum amount of time (for anytime solutions) that needs to be spent before its execution is terminated in order to get meaningful results.

- **Organizational Knowledge** Contains information about which other agents belong to the organization. This knowledge changes dynamically as a result of reorganization.

Apart from this, an agent has the following major functional components:

- **Sensor and Communication Processor** Senses the environment and handles message traffic with other agents.

- **Event Monitor** Checks for conditions that could result in dropping an existing intention and informs the override mechanism if they are satisfied. Otherwise, it generates a new objective and passes it on to the planner.

- **Override Mechanism** Drops intentions that are decommited or for which motivation no longer exists. When an agent receives information from its associates about decommitment or lack of motivation for a joint goal, the override mechanism drops the corresponding intention.

- **Planner:** Takes the new objective and determines whether it can be met and if so how. This is done on the basis of the recipe library and current intentions. The output of this is the agent's desire.
Inconsistency Revolver. For local activities it checks whether the desire is consistent with the current intentions. It then resolves inconsistency if any (on the basis of priority) and forms a new intention.

Contract Processor. For social goals the contract processor finds suitable team members that can carry out the goals individual actions. It then forms a joint intention for the goal after ensuring consistency of its actions with existing intentions.

Goal processor: Takes individual intentions and executes them. Intentions represent information about which task is to be executed and when. The goal processor is assumed to have the knowledge required to execute individual tasks specified in an intention. In order to accommodate anytime algorithms, a monitor in the goal processor continuously keeps track of the time for starting the next goal. It terminates the execution of the present goal when the time to start next goal arrives. Our protocol ensures that by this time the current goal has got its minimum chunk and has an acceptable result.

Agents in an organization receive a stream of time-constrained problem solving requests from the environment and from other agents in the organization. The objective of the event monitor process is to identify the following situations:

A new objective is raised

For instance, in a process control system, an agent may detect a fault and therefore should initiate the diagnosis process.

An event which is related to a local action occurs

An agent may be waiting for an acquaintance to provide a particular piece of information, when this event occurs the agent can continue with its processing.

An existing commitment is overridden.
Commitments are not irrevocable, therefore an agent must detect events which invalidate commitments so that it does not pursue fruitless activities.

Assume that an event, which signifies the need for fresh activity, is detected by the event monitor process. This new objective serves as input to the planner process, which determines whether it should be met and if so how. When deciding whether to adopt a new objective, the agent must consider its library of recipes and its current intentions. This allows the agent to determine whether the objective can be satisfied locally, or whether it necessitates social activity. Existing intentions must be taken into consideration while doing this, because they reflect activities the agent has already committed to. The output of the planner is a desire to pursue the objective locally, or to pursue it in a collaborative (social) fashion.

If the desire is to pursue the objective locally, it must ensure that the new intention is compatible with the existing ones. Compatibility means that it does not conflict with anything the agent has already committed itself to. For instance, for an agent capable of working on one task at a time, the decision for performing task t2 from time 5 to time 10 is incompatible with an earlier intention to perform a different task t1 from time 8 to time 15. If there are no inconsistencies, the new goal is added to the list of individual intentions and the agent commits itself to performing it. The goal processor then executes the tasks specified in these individual intentions.

If the new intention conflicts with the existing intentions, the inconsistency must be resolved; either by modifying the existing commitments or by altering the new intention so that it is no longer in conflict. An important consideration in such situations is the agent's preferences or desires. If the new goal is less important (less desirable) than existing ones, then it should be the one which is modified, conversely, if it is more desirable then it is the existing one which should be adapted. As a result of this modification, the less desirable tasks may not be able to complete as per their time schedule. Such goals are decommitted. Whenever an agent decommits...
Chapter 4 Organization and Agent Architecture in TRACE

Figure 4.3 Agent architecture in TRACE
a goal, it informs all other team members so that the entire team always remains coordinated.

If the planner decides that the objective can be met collaboratively, then the contract processor finds team members and establishes a joint intention to achieve the objective, using the task allocation protocol described in Chapter 5. The establishment of joint intention means that a group of agents agree to work together to achieve a common goal and that for the duration of this activity they will obey the responsibility code of conduct (explained in Chapter 2).

Joint intentions cannot be executed directly. Their role is to serve as a problem-solving context, which binds the actions of multiple agents together. It is the team members who have the ability to act and hence only individual intentions are directly related to actions. However, there is a causal link between the individual and joint intentions - each team member would be expected to adopt at least one individual intention as a consequence of its participation in a joint goal. Also, there must be consistency between the two representations. For example, if an individual has an intention to perform task t1 from time 10 to 20, this must be consistent with any of the related actions in the joint intention.

The task allocation process requires the other agents of the organization to perform local reasoning to fit the primitive actions in with their existing commitments.

Though the proposed architecture shares with the deliberative approach [3,94] the basic idea that belief, desire and intentions be represented explicitly. Reactive features [105, 106] can easily be introduced in our architecture. Normally, the planner process has a goal to achieve, for which it obtains a plan from the recipe library. The inconsistency resolver then checks for temporal compatibility of the new intention with the preexisting ones and resolves conflicts if any, based on priority among intentions. The resulting intentions are then passed on to the goal processor for execution. In order to incorporate reactive behavior (for unpredictable
environments), high priority goals which require immediate attention need to be achieved quickly, without spending time on planning. For this to happen such goals can be identified and an action for it stored in the event monitor. When the event monitor encounters any of these goals, it passes the request on to the inconsistency resolver to accommodate the high priority goal. Thus an intention can be generated for very critical activities without letting it go through the planner. In this way new-intentions can prevail on preexisting ones and reactively modify the agent's course of actions. A hybrid approach is therefore attained, where agent behavior is reactive or deliberative depending on the actual situation.

### 4.4 Conclusions

This chapter proposed an organization for the multi-agent system as well as the individual agent architecture. TRACE based applications are intended for use in unpredictable environments. TRACE agent architecture therefore allows agents to

i) Adapt to unpredictable changes in problem solving environment (by keeping its beliefs and goals always consistent with the latest information that it receives from the environment/other agents)

ii) Exhibit team rationality by means of the joint intention representation and the override mechanism. When the conditions for executing an intention no longer exist, the intention can be dropped through the override mechanism and fellow team members can be informed of this fact. This aids in time constrained problem solving by preventing agents from pursuing fruitless activities and reducing the amount of wasted effort.
iii) Focus on higher priority tasks. When an inconsistency is detected between an existing intention and a new one, it is resolved by the inconsistency resolver in favor of the higher priority intention.

In addition to this, the multi-agent system organization facilitates cooperative problem solving among agents within an organization, and allows the multi-agent system to:

i) adapt to changes in load by diverting resources where they are needed most,

ii) add new agents for problem solving in an incremental manner,

and thereby reorganize it dynamically. The reorganization process is described in detail in Chapter 6.